Modelling and mitigating Software Implementation Project Risks with Proposed Mining Technique

Abdelrafe Elzamly*[1], Burairah Hussin[2]

[1,2]Information and Communication Technology, University Technical Malaysia Malaka (UTeM)
Fakulti Teknologi Maklumat & Komunikasi, Universiti Teknikal Malaysia Melaka Locked Bag 1752, Durian Tunggal Post Office 76109 Durian Tunggal, Melaka Malaysia

[1]Department of Computer Science, Faculty of Applied Sciences, Al-Aqsa University, Gaza, Palestine, P.O.BOX: 4051

*[1]abd_elzamly@yahoo.com; [2]burairah@utem.edu.my

Received Apr 5, 2014; Accepted Jun 16, 2014; Published Jun 27, 2014

Abstract

The aim of this paper is to present new mining technique that uses the fuzzy multiple regression analysis modelling techniques to identify the risk management techniques that are effective in reducing the occurrence of each software implementation risks. Top ten software implementation risk factors and the best thirty risk management techniques were presented to respondents. As a result, we have shown that top ten software implementation risk factors can be reduced by risk management techniques according to mining techniques. This study will guide software managers to apply software risk management practices with real world software development organizations and verify the effectiveness of the new techniques and approaches on a software project. It is hoped that this will enable software managers improve their decision to increase the probability of software project success.

Keywords

Software Risk Management; Software Implementation Risks; Risk Management Techniques; Mining Technique

Introduction

Despite much research and progress in the area of software project management, software development projects still fail to deliver acceptable systems on time and within budget. According to (Yassin, 2010), identifying the risks that facing software projects and reasons behind their failure has haunted project managers, software industry consultants and academician for a long time. Therefore, management is still unable to effectively manage the risks involved in these software projects. Due to the involvement of risk management in monitoring the success of a software project, analyzing potential risks and making decisions about what to do with potential risks, the risk management is considered the planned control of risk. In addition, risk is an uncertainty that can have a negative or positive effect on meeting project objectives. This study incorporates between risk management approach and software development life cycle to mitigate software failure in implementation phase. Risk management is the process of identifying, analyzing, and controlling risk throughout the lifecycle of a software project to meet the software project objectives (Schwalbe, 2010). Clearly, the success or failure of software projects is generally assessed with dimensions such as budget, schedule, and quality (Miler, 2005). In this paper, we identify software implementation risk factors and risk management techniques on a large set that are guided software project managers to mitigate risks in a software project. According to (Hoffer, George, & Valacich, 2011), Software Development Life Cycle (SDLC) is the process of creating and methodologies that can use to develop a software project which including phases as planning, analysis, design, implementation and maintenance. In addition, we focused on Implementation phase: It involves the actual construction and installation of a system.

The objective of this paper is: To identify software implementation risk factors and risk management

techniques is considered from various literatures, modelling relationship between the software implementation risk factors and risk management techniques is develop using fuzzy multiple regression analysis.

Literature Review

There is no structure way of managing software risk, project manager using their experience, opinion and self-judgment to mitigate risk. Hence, techniques or model to mitigate software risks in a software project is necessary. In the literature, many considered on mitigate risk by qualitative and quantitative techniques, but rarely combine between software development life cycle and risk management based on mining techniques to mitigate software risks. In addition, a few authors combine between software risk factors and risk management techniques to reduce risks in SDLC. However (Khanfar et al., 2008), used chi-square ($\chi 2$) to mitigate risks in a software project by using control factors. And proposed new techniques the regression test and effect size test to manage the risks in a software project (Elzamly & Hussin, 2011b). Furthermore, the new stepwise regression technique used to manage the risks in a software project (Elzamly & Hussin, 2013a). Indeed, the multiple regression analysis techniques with fuzzy concepts is used to mitigate the risks in a software project in design phase (Elzamly & Hussin, 2013b). Also the fuzzy multiple regression analysis is used to reduce the risks in a software project in planning phase (Elzamly & Hussin, 2014b). Oracle corporation described risk management solutions enable a standardized approach for identifying risk, assessing risk and mitigating risk throughout the software project lifecycle (Oracle, 2010). Previous studies had shown that risk mitigation in software project can be classified by 3 categories such as qualitative, quantitative, and mining approaches. Mining approach is a new way of identifying risk from data that create relationships between data and find the optimum result from them. This includes techniques such as simulation analysis, fuzzy logic models, fuzzy multiple regression, neural network models, genetic algorithm, and heuristic algorithm. The goal of mining techniques is predicted to select the best model based on modelling and their prediction accuracy to mitigate risks. The new framework software risk management methodology proposed for successful software project that including 5 phases such as risk identification, risk analysis and evaluation, risk treatment, risk controlling, risk communication and documentation for software development life cycle which relied on three categories techniques as qualitative analysis, quantitative analysis and mining analysis to meet the goals (Elzamly & Hussin, 2014a).

Top 10 Software Implementation Risk Factors:

We displayed the top software implementation risk factors that most common used by researchers when studying the software risk in software projects in Table 1. However, the list consists of the 10 most serious risks to a project ranked from one to ten, each risk's status, and the plan for addressing each risk. These factors need to be addressed and thereafter need to be controlled. In this section the top software risk associated with implementation phase is discussed. These software implementation project risks are:

Risk 01: Failure to Gain User Commitment.

According to (Boehm, 1991; Elzamly and Hussin, 2011b; Jones, 2010; Keil et al., 2002; Khanfar et al., 2008; Kweku Ewusi-Mensah, 2003; Nakatsu and Iacovou, 2009; Schmidt et al., 2001; Tesch et al., 2007), failure to gain user commitment is the major risk during implementation phase. It is defined lack of client responsibility by the project manager rather than on the users (Schmidt et al., 2001). It is viewed as critical because it helps ensure that key users are actively involved in the requirements determination process, and it creates a sense of ownership, thereby minimizing the risk that the system (Keil, Cule, Lyytinen, & Schmidt, 1998).

Risk 02: Personnel Shortfalls.

According to Boehm's top 10 survey 2002, 2006-2007 software risks (Boehm, 1991, 2007; Dash and Dash, 2010; Jalote, 2008; Schmidt et al., 2001; Surie, 2004), the issue of highlight personnel shortfalls, this is defined as involvement fewer people than necessary or not having people with specific skills or qualified personnel that a software project might require (Jalote, 2008; Ropponen and Lyytinen, 2000). Personnel shortfalls will lead difficultly to estimate the size or cost of a software project (Galorath and Evans, 2006).

Risk 03: Failure to Utilize a Phased Delivery Approach.

However (Aritua, Smith, & Bower, 2011; Pandian, 2007), stated that inaccurate sizing of deliverable in software project is risk need to consider. Hallows

described the delivery risks such as response time of the system is not sufficient during implementation phase (Hallows, 2005). Thus, it is important to adapt a phased delivery approach to ensure that users are aware of when each module is delivered (Cliff Mitchell, 2011; Elzamly and Hussin, 2011b; Khanfar et al., 2008; Schmidt et al., 2001).

Risk 04: Too Little Attention to Breaking Development and Implementation into Manageable Steps.

According to (Bronte-Stewart, 2009; Cliff Mitchell, 2011; Elzamly and Hussin, 2011b; Sumner, 2000), the too little attention is put to break the development and implementation into a manageable steps. House of Commons (2009) reported that is because it is reasonable for a software project that is large and complex software system (Anonymous, 2009). Thus, it may lead to insufficient implementation for software project late delivering or failing to deliver at all.

Risk 05: Inadequate Training Team Members.

According to (Aloini et al., 2007; Aziz and Salleh, 2011; Elzamly and Hussin, 2011a, 2011b; Han and Huang, 2007; Huang and Han, 2008; Jalote, 2002; Surie, 2004), identified that inadequate training software development members is important in implementation stage. The training needs may arise directly from the knowledge and skills needed to operate the new technology (Buckley and Caple, 2009). Consequently, inadequate training may also cause the team to use the tools and techniques incorrectly or inefficiently, creating performance or functionality problems (Tayntor, 2006).

Risk 06: Inadequacy of Source Code Comments.

According to (Chen and Huang, 2009; Jones, 2008), reported that complexity and source code without good programming comment is a disaster. This is because every software project should be examined in a baseline analysis should be evaluated for the complexity comments do helps developers. The most commonly issues as non-comment lines in a program's source code (Galorath and Evans, 2006). Thereby, due to the low quality of measuring the software maintainability, such as no integrated use of source code, comments ratio, and specifications will lead implementation difficulties.

Risk 07: Inadequate Test Cases and Generate Test Data.

According to (Addison, 2003; Lientz and Larssen, 2006; Pandian, 2007), lack of time to generate test data do contribute to the missing reusable test plans and test case (Addison, 2003; Pandian, 2007). Test data is important to aid testing process during implementation phase (Dustin, 2002; Naik and Priyadarshi, 2008; Stephens and Doug, 2010). Failure to have the sufficient test data, will lead to potential risk to the application.

Risk 08: Real-time Performance Shortfalls.

According to (Boehm, 1991; Ropponen and Lyytinen, 2000), real-time performance shortfalls in software project is a result of the system (Ropponen and Lyytinen, 2000). The major challenges in this issue is building what is the real time performance required (Madhavji, Fernandez, & Perry, 2006). Thereby, it is necessary as a performance criteria to protect the software failure (Hillary, 2005; Surie, 2004).

TABLE 1 ILLUSTRATE TOP TEN SOFTWARE IMPLEMENTATION RISK FACTORS BASED ON RESEARCHERS

Phase	No	Software Implementation risk factors	Frequency
Implementation	1	Failure to gain user commitment (Addison & Vallabh, 2002; Keil et al., 2002; Kweku Ewusi-Mensah, 2003; Schmidt et al., 2001)	5
	2	Personnel shortfalls (Boehm, 1991, 2002a, 2002b, 2007).	4
	3	Failure to utilize a phased delivery approach (Aritua et al., 2011; Khanfar et al., 2008)	2
	4	Too little attention to breaking development and implementation into manageable steps (Cliff Mitchell, 2011; Sumner, 2000).	2
	5	Inadequate training team members (Aloini et al., 2007).	1
	6	Inadequacy of source code comments (Chen & Huang, 2009).	1
	7	Inadequate test cases and generate test data (Addison, 2003).	1
	8	Real-time performance shortfalls (Boehm, 1991, 2002b).	1
	9	Test case design and Unit-level testing turns out very difficult (Elzamly & Hussin, 2011a).	1
	10	Lack of adherence to programming standards (Chen & Huang, 2009).	1
		Total frequency	**19**

Risk 09: Test Case Design and Unit-level Testing Turns out Very Difficult.

According to (Elzamly and Hussin, 2011b; Stephens and Doug, 2010), test case design and unit level to create test case for each activity and decision nodes is critical during implementation phase. Therefore (Laplante and Ovaska, 2005), once the test cases were designed, it's really easier to write the code and running the process. Thereby, unit testing is to isolate each part of the program and potential show that individual parts are incorrect in terms of requirements and functionality (Jalote, 2008).

Risk 10: Lack of Adherence to Programming Standards.

According to (Chen and Huang, 2009; Melnick and Everitt, 2008), reported that adherence to programming standards (e.g. the rules of structured programming) is a critical matter. Indeed, intentional deviation from specification and the violation of established programming standards more often leads to minor errors during the maintenance phase this errors will because more serious error (Schulmeyer, 2008; Vaidyanathan and Devaraj, 2003).

Risk Management Techniques

Through reading the existing literature on software risk management, we listed the most common thirty risk management techniques that are considered important in mitigating the software implementation risks identified; in the study, we summarize 30 risk management techniques in mitigating risk as follows:

C1: Using of requirements scrubbing, C2: Stabilizing requirements and specifications as early as possible, C3: Assessing cost and scheduling the impact of each change to requirements and specifications, C4: Develop prototyping and have the requirements reviewed by the client, C5: Developing and adhering a software project plan, C6: Implementing and following a communication plan, C7: Developing contingency plans to cope with staffing problems, C8: Assigning responsibilities to team members and rotate jobs, C9: Have team-building sessions, C10: Reviewing and communicating progress to date and setting objectives for the next phase, C11: Dividing the software project into controllable portions, C12: Reusable source code and interface methods, C13: Reusable test plans and test cases, C14: Reusable database and data mining structures, C15: Reusable user documents early, C16: Implementing/Utilizing automated version control tools, C17: Implement/Utilize benchmarking and tools of technical analysis, C18: Creating and analyzing process by simulation and modeling, C19: Provide scenarios methods and using of the reference checking, C20: Involving management during the entire software project lifecycle, C21: Including formal and periodic risk assessment, C22: Utilizing change control board and exercise quality change control practices, C23: Educating users on the impact of changes during the software project, C24: Ensuring that quality-factor deliverables and task analysis, C25: Avoiding having too many new functions on software projects, C26: Incremental development (deferring changes to later increments), C27: Combining internal evaluations by external reviews, C28: Maintain proper documentation of each individual's work, C29: Provide training in the new technology and organize domain knowledge training, C30: Participating users during the entire software project lifecycle.

Empirical Strategy

Data collection was achieved through the use of a structured questionnaire in estimating the quality of software. Top ten software Implementation risk factors and thirty risk management techniques were presented to respondents. The method of sample selection referred to as 'snowball' and distribution personal regular sampling was used. This procedure is appropriate when members of homogeneous groups (such as software project managers, IT managers) are difficult to locate. The seventy six software project managers have participated in this study. Respondents were presented with various questions, which used scale 1-7. All questions in software risk factors were measured by a 7-point Likert scale from unimportant to extremely important and risk management techniques were measured by a seven-point scale from never to always. The data is collected from various to IT manager, software project managers from software development organizations in Palestine. In order to find the relation among software implementation risks and risk management techniques, we introduce fuzzy multiple regression to mitigate software project implementation risks.

Regression Analysis Model with Fuzzy Concepts

Fuzzy multiple regression analysis is an extension of the traditional regression analysis in which some elements of the models are represented by fuzzy numbers (Dom, Abidin, Kareem, Ismail, & Daud, 2012). On the other words, fuzzy multiple regression

analysis in that response variable is fuzzy variable and part of the covariates are crisp variables (Lin, Zhuang, & Huang, 2012). However, identify the various data types that may appear in a questionnaire. Thus, we introduce the survey questionnaire data mining risk and define the rule patterns that can be mined from survey questionnaire data (Chen & Huang, 2009). Therefore, we must extend the crisp association rules to fuzzy association rules from questionnaire data.

Fuzzy Concepts with Membership Function

Fuzzy concepts help to find the deviation of each data from goodness equation, so we define a normal distribution membership function as follow (Marza & Seyyedi, 2009):

$$U_i = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{Yi-\mu}{\sigma}\right)^2} \quad (1)$$

Where:
μ = Average of sample points and
σ = Square root of variance math

If we add fuzzy domain to regression technique, the effect of discrete data points on the goodness result will be reduced and the effect of concentrated data points on the fitness result will be enhanced. Indeed, a membership function is a curve that defines how each point in the input space is mapped to a membership value between 0 and 1 (Dom et al., 2012).

Fuzzy Parameters

A group of these equations to obtain the fuzzy parameters are provided as (Gu, Song, & Xiao, 2006; Popescu & Giuclea, 2007):

$$\begin{aligned}
&s11\ b1 + s12b2 + \dots\dots\dots\dots + s1kbk = s1y\\
&s21\ b1 + s22b2 + \dots\dots\dots\dots + s2kbk = s2y\\
&s31\ b1 + s32b2 + \dots\dots\dots\dots + s3kbk = s3y\\
&s41\ b1 + s42b2 + \dots\dots\dots\dots + s4kbk = s4y\\
&s51\ b1 + s52b2 + \dots\dots\dots\dots + s5kbk = s5y\\
&\dots\dots\dots\dots\dots\\
&sk1\ b1 + sk2b2 + \dots\dots\dots\dots + skkbk = sky
\end{aligned} \quad (2)$$

Here

$$sij = \sum u \sum uXiXj - \sum uXi \sum uXj$$

$$,and$$

$$siy = \sum u \sum uXiy - \sum uXi \sum uy$$

According to the group of equations, first we can obtain the values of variables b1, b2… bk and finally b0 is gained by:

$$b_0 = \frac{\sum uy}{\sum u} - b1\frac{\sum ux1}{\sum u} - b2\frac{\sum ux2}{\sum u} - \dots - bk\frac{\sum uxk}{\sum u} \quad (3)$$

Relationships Between Software Implementation Risks and Risk Management Techniques:

Fuzzy multiple Regression technique was performed on the data to determine whether there were significant relationships between risk management techniques and software implementation risks and to compare the risk management techniques to each of the risk factors to identify if they are effective in mitigating the occurrence of each software implementation risk. Relationships between risks and risk management techniques, which were significant and insignificant, any risk management technique is no significant; we are not reported according to the best model.

R1: Risk of 'Failure to Gain User Commitment' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the α = 0.05 level of significance, so the control 3 has an impact on the risk 1. The value of R^2 is 0.0119; hence this interprets as a percentage of 1.194 % from the dependent variable of risk 1. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk1} = 4.108481023 + 0.085604511*C3 \quad (4)$$

R2: Risk of 'Personnel Shortfalls' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the α = 0.05 level of significance, so the control 3 has an impact on the risk 2. The value of R^2 is 0.0717; hence this interprets as a percentage of 7.17 % from the dependent variable of risk 2. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk 2} = 4.013763437 + 0.44022812*C3 \quad (5)$$

R3: Risk of 'Failure to Utilize a Phased Delivery Approach' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the α = 0.05 level of significance, so the control 7 has an impact on the risk 3. The value of R^2 is 0.032; hence this interprets as a percentage of 3.20 % from the dependent variable of risk 3. According to the fuzzy concepts in multiple regression

analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk3} = 3.842352646 + 0.165843191*C7 \qquad (6)$$

R4: Risk of 'Too Little Attention to Breaking Development and Implementation into Manageable Steps' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the $\alpha = 0.05$, so the control 3 and 29 have an impact on the risk 4. The value of R^2 is 0.15; hence this interprets as a percentage of 15.7 % from the dependent variable of risk 4. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk4} = 3.341376335 + 0.142586809*C3 + 0.154868976*C29 \qquad (7)$$

R5: Risk of 'Inadequate Training Team Members' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the $\alpha = 0.05$ level of significance, so control 3 and 29 have an impact on the risk 5. The value of R^2 is 0.0809; hence this interprets as a percentage of 8.09 % from the dependent variable of risk 5. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk5} = 2.74243438 + 0.21196221*C3 + 0.123098227*C29 \qquad (8)$$

R6: Risk of 'Inadequacy of Source Code Comments' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the $\alpha = 0.05$ level of significance, so the control 4 has an impact on the risk 6. The value of R^2 is 0.034; hence this interprets as a percentage of 3.4 % from the dependent variable of risk 6. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk6}=3.725+ 0.190* C4 \qquad (9)$$

R7: Risk of 'Inadequate Test Cases and Generate Test Data' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the $\alpha = 0.05$ level of significance, so the control 3 has an impact on the risk 7. The value of R^2 is 0.03228; hence this interprets as a percentage of 3.22% from the dependent variable of risk 7. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{FuzzyRisk7}= 3.8199687646 + 0.183544213*C3 \qquad (10)$$

R8: Risk of 'Real-Time Performance Shortfalls' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the $\alpha = 0.05$ level of significance, so the control 3, and 29 have an impact on the risk 8. The value of R^2 is 0.142. This interprets as a percentage of 14.2 % from the dependent variable of risk 8. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk8} = 2.30265414 + 0.151639036*C3 + 0.246665034*C29 \qquad (11)$$

R9: Risk of 'Test Case Design and Unit-Level Testing Turns out Very Difficult' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the $\alpha = 0.05$ level of significance, so the control 2 and 28 have an impact on the risk 9. The value of R^2 is 0.0786; hence this interprets as a percentage of 7.86 % from the dependent variable of risk 9. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{Fuzzy Risk9} = 2.630149718 + 0.242144059*C2 + 0.12748371*C28 \qquad (12)$$

R10: Risk of 'Lack of Adherence to Programming Standards' Compared to 30 Controls.

The results show that the significant value is less than the assumed value at the $\alpha = 0.05$ level of significance, so the control 1, 7, and 29 have an impact on the risk 10. The value of R^2 is 0.1011; hence this interprets as a percentage of 10.11 % from the dependent variable of risk 10. According to the fuzzy concepts in multiple regression analysis to produce a fuzzy multiple regression model, by solving these equations above, the final fuzzy equation:

$$\text{FuzzyRisk10} = 3.665033759 + 0.043015874*C1 + 0.170747137*C7 + 0.183660567*C29 \qquad (13)$$

TABLE 2 SOFTWARE RISK FACTORS ARE MITIGATED BY RISK MANAGEMENT TECHNIQUES

No	Software Implementation Risk Factors	Risk Management Techniques
1	Failure to gain user commitment.	C3: Assessing cost and scheduling the impact of each change to requirements and specifications.
2	Personnel shortfalls.	C3: Assessing cost and scheduling the impact of each change to requirements and specifications.
3	Failure to utilize a phased delivery approach.	C7: Developing contingency plans to cope with staffing problems.
4	Too little attention to breaking development and implementation into manageable steps.	C3: Assessing cost and scheduling the impact of each change to requirements and specifications, C29: Provide training in the new technology and organize domain knowledge training.
5	Inadequate training team members.	C3: Assessing cost and scheduling the impact of each change to requirements and specifications, C29: Provide training in the new technology and organize domain knowledge training.
6	Inadequacy of source code comments.	C4: Develop prototyping and have the requirements reviewed by the client.
7	Inadequate test cases and generate test data.	C3:Assessing cost and scheduling the impact of each change to requirements and specifications
8	Real-time performance shortfalls.	C29: Provide training in the new technology and organize domain knowledge training, C3: Assessing cost and scheduling the impact of each change to requirements and specifications.
9	Test case design and Unit-level testing turns out very difficult.	C2: Stabilizing requirements and specifications as early as possible, C28: Maintain proper documentation of each individual's work.
10	Lack of adherence to programming standards.	C7: Developing contingency plans to cope with staffing problems, C29: Provide training in the new technology and organize domain knowledge training, C1: Using of requirements scrubbing.

Software Implementation Risks Identification Checklists and Risk Management Techniques:

Table 2 shows software implementation risk factors identification checklist with risk management techniques based on a questionnaire of experienced software project managers. He can use the checklist on software projects to identify and mitigate risk factors on software projects lifecycle by risk management techniques.

Conclusion

The concern of our paper is the managing risks of software projects in the implementation phase. The results show that all risks in software projects were important in software project manager's perspective, whereas all controls are used most of the time, and often. These tests were performed using fuzzy multiple regression analysis to compare the controls to each of the implementation software risk factors to determine if they are effective in mitigating the occurrence of each implementation software risks. Relationships between risks and risk management techniques, which were significant and insignificant, any control is no significant, we are not reported. We used multiple regression analysis techniques with fuzzy concepts by MATLAB 7.12.0 (R2011a), wolfram mathematic 9.0. However, we referred the control factors were mitigated on risk implementation factors in Table 2. Through the results, we found out that some controls haven't impacted, so the important controls should be considered by the software development companies in Palestinian. In addition, we cannot obtain historical data from database by using some techniques. Ror the reason that, some of software project managers didn’t give us a historical template to follow up software risks. In addition, there is no software clearly to compute the fuzzy regression analysis and combine between linear and nonlinear technique. As future work, we will intend to apply these study results on a real-world software project to verify the effectiveness of the new techniques and approach on a software project. Likewise, we can use other nonlinear techniques to find the relation between software risk and risk management techniques that we believe can contain better result. Also, we could hybridize the techniques with other artificial intelligence approach to improve the models.

ACKNOWLEDGEMENT

This study is supported by Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka (UTeM) and Al-Aqsa University, Gaza, Palestine.

REFERENCES

Addison, T. (2003). E-commerce project development risks:

Evidence from a Delphi survey. *International Journal of Information Management*, *23*(2003), 25–40.

Addison, T., & Vallabh, S. (2002). Controlling Software Project Risks – an Empirical Study of Methods used by Experienced Project Managers. In *Proceedings of SAICSIT* (pp. 128 – 140).

Aloini, D., Dulmin, R., & Mininno, V. (2007). Risk management in ERP project introduction: Review of the literature. *Information & Management*, *44*(6), 547–567.

Anonymous. (2009). *The National Offender Management Information System* (p. 41). Parliament: House of Commons: Committee of Public.

Aritua, B., Smith, N. J., & Bower, D. (2011). What risks are common to or amplified in programmes: Evidence from UK public sector infrastructure schemes. *International Journal of Project Management*, *29*(3), 303–312.

Aziz, M., & Salleh, H. (2011). People Critical Success Factors of IT/IS Implementation: Malaysian Perspectives. *World Academy of Science, Engineering, and Technology*, *80*(56), 75–82.

Boehm, B. (1991). Software Risk Management: Principles and Practices. *IEEE Software*, (January), 1–10.

Boehm, B. (2002a). Boehm's Top 10 2002 - Software Risks. *The USC Center for Systems and Software Engineering*. Retrieved February 28, 2012, from http://csse.usc.edu/BoehmsTop10/2006/survey.php

Boehm, B. (2002b). *Software Risk Management: Overview and Recent Developments* (Vol. 2002, p. 90).

Boehm, B. (2007). *USC CSSE Workshop Overview: Top 3 Software-Intensive Systems Risk Items* (p. 34).

Bronte-Stewart, M. (2009). Risk Estimation From Technology Project Failure. In *4th European Conference on Management of Technology* (pp. 1–19).

Buckley, R., & Caple, J. (2009). *The Theory & Practice of Training* (6th ed., p. 369). kogan page London and Philadelphia.

Chen, J., & Huang, S. (2009). An empirical analysis of the impact of software development problem factors on software maintainability. *The Journal of Systems and Software*, *82*(6), 981–992.

Cliff Mitchell. (2011). Why don't we learn from our project failures. *University of Manchester*. Retrieved March 01, 2012, from https://tm.mbs.ac.uk/SearchResults/tabid/56/ArticleID/72/Why-don%E2%80%99t-we-learn-from-our-project-failures.aspx

Dash, R., & Dash, R. (2010). Risk Assessment Techniques for Software Development. *European Journal of Scientific Research*, *42*(4), 629–636.

Dom, R., Abidin, B., Kareem, S., Ismail, S., & Daud, N. (2012). Determining the Critical Success Factors of Oral Cancer Susceptibility Prediction in Malaysia Using Fuzzy Models. *Sains Malaysiana*, *41*(5), 633–640.

Dustin, E. (2002). *Effective Software Testing: 50 Specific Ways to Improve Your Testing* (p. 203). Addison-Wesley Professional.

Elzamly, A., & Hussin, B. (2011a). Estimating Quality-Affecting Risks in Software Projects. *International Management Review, American Scholars Press*, *7*(2), 66–83.

Elzamly, A., & Hussin, B. (2011b). Managing Software Project Risks with Proposed Regression Model Techniques and Effect Size Technique. *International Review on Computers and Software (I.RE.CO.S.)*, *6*(2), 250–263.

Elzamly, A., & Hussin, B. (2013a). Managing Software Project Risks (Implementation Phase) with Proposed Stepwise Regression Analysis Techniques. *International Journal on Information Technology (IREIT)*, *1*(4), 300–312.

Elzamly, A., & Hussin, B. (2013b). Managing Software Project Risks (Design Phase) with Proposed Fuzzy Regression Analysis Techniques with Fuzzy Concepts. *International Review on Computers and Software (I.RE.CO.S.)*, *8*(11), 2601–2613.

Elzamly, A., & Hussin, B. (2014a). An Enhancement of Framework Software Risk Management Methodology for Successful Software Development. *Journal of Theoretical and Applied Information Technology*, *62*(2), 410–423.

Elzamly, A., & Hussin, B. (2014b). Managing Software Project Risks (Planning Phase) with Proposed Fuzzy Regression Analysis Techniques with Fuzzy Concepts. *International Journal of Information and Computer Science (IJICS)*, *3*(2), 31–40. doi:10.14355/ijics.2014.0302.02

Galorath, D., & Evans, M. (2006). *Software Sizing, Estimation, and Risk Management*. *Distributed Computing* (p. 541). Auerbach Publications.

Gu, X., Song, G., & Xiao, L. (2006). Design of a Fuzzy

Decision-making Model and Its Application to Software Functional Size Measurement. In *International Conference on Computational Intelligence for Modelling Control and Automation, and International Conference on Intelligent Agents,Web Technologies and Internet Commerce (CIMCA-IAWTIC'06)* (pp. 199–205).

Hallows, J. (2005). *Information Systems Project Management: How to Deliver Function and Value in Information Technology Projects* (Second., p. 304). AMACOM.

Han, W., & Huang, S. (2007). An empirical analysis of risk components and performance on software projects. *Journal of Systems and Software*, *80*(1), 42–50.

Hillary, N. (2005). *Measuring Performance for Real-Time Systems*. *Measuring Performance for Real-Time Systems Freescale Semiconductor* (pp. 1–14).

Hoffer, J., George, J., & Valacich, J. (2011). *Modern Systems Analysis and Design* (6th ed., p. 575). Prentice Hall.

Huang, S., & Han, W. (2008). Exploring the relationship between software project duration and risk exposure: A cluster analysis. *Information & Management*, *45*(3), 175–182.

Jalote, P. (2002). *Software Project Management in Practice* (p. 288). Addison Wesley.

Jalote, P. (2008). *A Concise Introduction to Software Engineering* (p. 279). Springer-Verlag London Limited.

Jones, C. (2008). *Applied Software Measurement Global Analysis of Productivity and Quality* (Third., p. 662). McGraw-Hill Companies.

Jones, C. (2010). *Software Engineering Best Practices:Lessons from Successful Projects in the Top Companies* (p. 688). McGraw-Hill Companies.

Keil, M., Cule, P., Lyytinen, K., & Schmidt, R. (1998). A framework for Identifying Software Project Risks. *Communications of the ACM*, *41*(11), 76–83.

Keil, M., Tiwana, A., & Bush, A. (2002). Reconciling user and project manager perceptions of IT project risk: A Delphi study. *Information Systems Journal*, *12*(2), 103–119.

Khanfar, K., Elzamly, A., Al-Ahmad, W., El-Qawasmeh, E., Alsamara, K., & Abuleil, S. (2008). Managing Software Project Risks with the Chi-Square Technique. *International Management Review*, *4*(2), 18–29.

Kweku Ewusi-Mensah. (2003). *Software Development Failures: Anatomy of Abandoned Projects* (p. 276). The MIT Press.

Laplante, P., & Ovaska, S. (2005). Chapter 1: Introduction. In *Real-time Systems Design and analysis* (4th ed., p. 53). Wiley-Blackwell.

Lientz, B. P., & Larssen, L. (2006). *Risk Management for IT Projects: How to Deal with Over 150 Issues and Risks* (p. 331). Elsevier Inc.

Lin, J., Zhuang, Q., & Huang, C. (2012). Fuzzy Statistical Analysis of Multiple Regression with Crisp and Fuzzy Covariates and Applications in Analyzing Economic Data of China important. *Computational Economics*, *39*(1), 29–49.

Madhavji, N., Fernandez, J., & Perry, D. (2006). *Software Evolation and Feedback* (p. 612). John Wiley & Sons Ltd.

Marza, V., & Seyyedi, M. (2009). Fuzzy Multiple Regression Model for Estimating Software Development Time, *1*(2), 31–34.

Melnick, E., & Everitt, B. (2008). *Encyclopedia of Quantitative Risk Analysis and Assessment*. (E. L. Melnick & B. S. Everitt, Eds.) (p. 2176). Chichester, UK: John Wiley & Sons, Ltd.

Miler, J. (2005). *A Method of Software Project Risk Identification and Analysis*. *Technology*. Gdansk University of Technology.

Naik, K., & Priyadarshi Tripathy. (2008). *Software Testing and Quality Assurance: Theory and Practice* (p. 648). Wiley-Spektrum.

Nakatsu, R., & Iacovou, C. (2009). A comparative study of important risk factors involved in offshore and domestic outsourcing of software development projects: A two-panel Delphi study. *Information & Management*, *46*(1), 57–68.

Oracle. (2010). *A Standardized Approach to Risk Improves Project Outcomes and Profitability*. *oracle white paper* (p. 7).

Pandian, C. R. (2007). *Applied software risk management: A guide for software project managers*. *Risk Management* (p. 246). Auerbach Publications is an imprint of the Taylor & Francis Group.

Popescu, C., & Giuclea, M. (2007). A model of multiple linear regression. *The Publishing House of the Romanian Academy*, *8*(2), 1–8.

Ropponen, J—., & Lyytinen, K. (2000). Components of software development risk: how to address them? A project manager survey. *IEEE Transactions on Software Engineering*, *26*(2), 98–112.

Schmidt, R., Lyytinen, K., Keil, M., & Cule, P. (2001).

Identifying Software Project Risks: An International Delphi Study. *Journal of Managemen Information Systems*, *17*(4), 5–36.

Schulmeyer, G. (2008). *Handbook of Software Quality Assurance. Quality* (Fourth., p. 464). ARTECH HOUSE, INC.

Schwalbe, K. (2010). *Information Technology Project Management* (sixth., p. 490). Course Technology, Cengage Learning.

Stephens, M., & Doug Rosenberg. (2010). *Design Driven Testing: Test Smarter, Not Harder* (p. 365). Matt Stephens and Doug Rosenberg.

Sumner, M. (2000). Risk factors in enterprise-wide/ERP projects. *Journal of Information Technology*, *15*(4), 317–327.

Surie, D. (2004). Evaluation and Integration of Risk Management in CMMI and ISO / IEC 15504. In *8th Student Conference in Computing Science, Umeå, Sweden* (p. 14).

Tayntor, C. (2006). *Successful Packaged Software Implementation. Changes* (p. 337). AUERBACH Publications, Taylor & Francis Group, LLC.

Tesch, D., Kloppenborg, T., & Frolick, M. (2007). IT Project Risk Factors: The Project Management Professionals Perspective. *Journal of Computer Information Systems*, *47*(4), 61–70.

Vaidyanathan, G., & Devaraj, S. (2003). A five-factor framework for analyzing online risks in e-businesses. *Communications of the ACM*, *46*(12), 354. doi:10.1145/953460.953522

Yassin, A. (2010). *Organizational Information Markets: Conceptual Foundation and an Approach for Software Project Risk Management*. University of South Florida.

Abdelrafe Elzamly is currently studying a Ph. D. in Software and Information Systems Engineering from Faculty of Information and Communication Technology at Technical University Malaysia Melaka (UTeM), Born in November 30, 1976, Gaza, Palestine. He received his B.Sc. degree computer in 1999 from Al-Aqsa University, Gaza, and his Master's degree in computer information system in 2006 from The University of Banking and Financial Sciences. He is working as lecturer in Computer Science at Al-Aqsa University from 1999 to 2014 as a full time and worked as lecturer at the Islamic University in Gaza from 1999 to 2007 as a part time. He also worked as a manager in The Mustafa Center for Studies and Scientific Research-Gaza from 2010 to 2012. His research of interest is risk management, quality software, software engineering, and data mining.

Burairah Hussin got a Ph. D. in Management Science – Condition Monitoring Modelling from University of Salford, UK in 2007. He received his M.Sc. Degree in Numerical Analysis and Programming from University of Dundee, UK in 1998. He received his B.Sc. Degree in Computer Science from University Technology Malaysia in 1996. He is currently working as associate professor in University Technical Malaysia Melaka (UTeM) and he is working as Research Manager at Centre for Advanced Computing Technology (C-ACT), Faculty of Information and Communication Technology at Technical University of Malaysia Melaka (UTeM). Also he worked as Deputy Dean (Research and Post Graduate), Faculty of Information and Communication Technology at Technical University of Malaysia Melaka (UTeM). His research interests are in data analysis, data mining, maintenance modelling, artificial intelligent, risk management, numerical analysis, and computer network advisor and development.